보안패치 자동분배를 위한 패치 DB 자동구성 방안

민동옥° 손태식 서정택 구원본 장정아 문종섭
고려대학교 정보보호기술연구센터
{eieshine°, tsshon, seojt, allclear_koo, eosjung, jsmoon}@cist.korea.ac.kr

Automatic Composition Database For Security Patch Auto-Distribution

Dong-Og Min° Tae-Shik Shon Jung-Taek Seo Won-Bon Koo Jung-Ah Jang Jong-Sub Moon
CIST, Korea University

요 약

근래 정보보안의 필요성이 대두됨에 따라 개인 사용자의 보안 지식이 다방면으로 필요하게 되었다. 특정 그룹에 속한 개인 사용자들의 보안패치나 백신등에 대한 지식이 비슷한 수준일 수 없으므로 보안패치나 백신등을 관리해 주는 솔루션이 필요하게 되었다. 이는 통합관제 센터나 혹은 보안패치 자동분배 솔루션을 사용하여 해결할 수 있으나, 시스템 관리자의 지속적인 패치 확인과 업데이트가 필요한 일이다. 시스템 관리자가 벤더를 검색하여 패치를 가져오는 방법은 지극히 주관적이고, 실수로 가져오지 못하거나 혹은 벤더의 업데이트 시간과 맞지 않아 긴급패치를 누락시키는 경우가 종종 발생한다. 또한, 자동분배 솔루션을 사용하는 네트워크의 병합이나 패치분배의 추가에 이렇다 할 대안이 없다. 이 논문에서 제안하는 패치 자동업데이트와 자동구성에 대한 방안은 패치의 누락을 미연에 방지하고, 네트워크의 병합등 패치분배 서버가 늘어나면서 생기는 문제점을 해결해준다.

1. 서 론

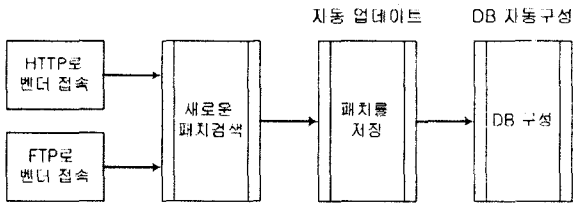
2003년 1.25대란을 통해 보안패치의 필요성과 중요성에 대해서는 모두 실감을 하였다. 그러나, 실질적으로 개개인의 사용자가 벤더들로부터 보안패치를 검색하고 꾸준히 업데이트 하는 것은 다소 무리가 있다. 이것을 해결하기 위해 보안패치 자동분배 솔루션을 통한 소수의 시스템관리자로부터 다수의 사용자에게 보안패치를 제공해주는 시스템을 도입하고 있는 추세이다. 그러나, 시스템관리자의 업무적 능력 한계와 벤더로부터 가져오는 패치의 주관적인 선택으로 인해 중요한 패치들이 누락되는 경우가 종종 발생한다. 이것은 벤더를 일일이 검사하여 필요한 패치인지를 테스트하고 적용시켜야 하는 시스템관리자의 역할이 너무 과다하기 때문에 발생하는 문제이다.

또한, 보안패치 솔루션을 적용하는 네트워크의 확장성을 고려해 두 개 이상의 네트워크를 병합하게 될 경우 패치분배 서버들의 패치들과 패치 데이터베이스가 동기화 되어야 할 필요가 있다.

이러한 문제들을 미연에 방지하고 네트워크가 확장되었을 때 패치서버와 패치 데이터베이스를 동기화하기 위해 각 벤더들을 자동으로 검사하여 패치를 수집하여 데이터베이스에 기록하고 패치 데이터베이스에 패치를 저장하는 보안패치 DB 자동구성 방안과 자동업데이트 방안을 제시하도록 한다.

2. 패치 DB 자동구성 시스템 설계

패치 DB를 자동구성하기 위해서는 크게 두 가지 부분으로 나뉘어지는데, 첫 번째가 패치를 벤더로부터 검색하여 가져오는 자동 업데이트 부분, 두 번째가 벤더로부터 가져온 패치를 접근하기 쉬운 방식으로 데이터베이스에 저장하는 DB 자동구성 부분이다.



[그림 1] 보안패치 DB 자동구성 시스템 구성도

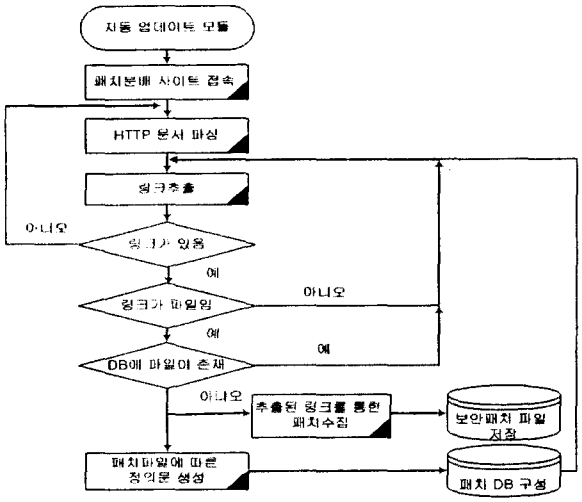
벤더가 패치정보를 서비스하는 방식은 크게 HTTP와 FTP가 있으며, 이 방식에 따라 벤더에 접속하는 방법과 벤더에 접속해서 패치를 검색하는 방법이 결정된다. 패치를 저장하고, 분배해 주는 패치 분배서버는 임의적으로 FTP로 서비스한다고 가정하였다.

3. 패치 자동 업데이트

자동업데이트 부분은 벤더가 제공하는 서비스의 형태에 따라 HTTP 접근과 FTP 접근으로 나뉜다. MicroSoft와 같이 패치정보를 웹사이트에 게시판형식으로 서비스는 벤더는 HTTP로 접근하여 게시물을 파싱하여 패치를 검색하고, RedHat과 같이 패치가 저장된 FTP 사이트를 열어서 서비스하는 벤더는 FTP로 접속해서 패치들을 검색한다.

3.1 HTTP 방식의 자동 업데이트

MicroSoft 혹은 Sun 사와 같이 패치에 대한 정보를 웹사이트의 게시판에 제공하는 벤더들은 HTTP로 웹사이트의 주소를 접근해 해당문서를 파싱하여 그 링크를 확보한 후 문서에 걸려있는 링크를 추적하여 패치를 수집하는 방법을 사용한다.



[그림 2] HTTP 자동구성 순서도

HTTP 게시판에 접속하여 기본문서를 파싱하기 시작하여 해당 링크들을 추출한다. 추출의 키워드는 'a href=' 와 같은 HTML 태그가 된다. 추출된 링크의 예는 다음과 같다.

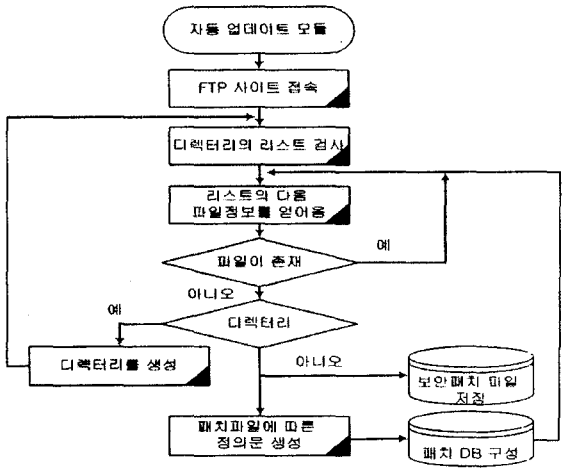
http://download.microsoft.com/download/2/8/9/28923ba1-0b28-4b20-9cf7-e3025b6335fa/WindowsXP-KB824146-x86-KOR.exe (1)

확장자가 exe파일이 아닌 다른 경로를 나타내는 링크에 대해서는 '다운로드' 혹은 'download' 와 같은 키워드를 이용하여 패치파일에 근접하게 찾아간다. 예시에서와 같이 확장자가 exe인 패치파일을 찾게되면, exe파일의 파일이름을 추출하여 정의문을 생성한다. 정의문을 이용한 쿼리를 DB에 보내 해당 파일이 DB에 존재하는지 확인한 후, 존재하지 않으면 생성된 정의문과 체크섬, 날짜, 패치 파일이 저장된 경로 등으로 패치 DB를 구성한다. 정의문에 대한 구체적인 내용은 4절 DB자동구성에서 알아본다.

3.2 FTP 방식의 자동 업데이트

RedHat과 같이 보안패치를 정리해 놓은 FTP사이트를 제공하는 벤더들은 FTP서버에 접속하여 FTP디렉토리와 패치 파일이름으로 해당 패치파일을 검색하여 패치를 수집하는 방법을 사용한다.

FTP 계정정보를 이용해 벤더의 FTP서버에 접속하여 디렉터리의 리스트를 얻어온다. 얻어온 리스트에서 순차적으로 파일존재, 디렉터리/파일 판별의 비교를 거친다. 새로운 디렉터리의 경우 디렉터리를 생성하고 생성된 디



[그림 3] FTP 자동구성 순서도

렉터리로 이동하여 검색을 진행한다. 새로운 패치 파일의 경우 서버에 패치파일을 저장하고 정의문, 체크섬, 날짜, 패치파일의 경로 등으로 패치 DB를 구성한다.

저장되는 패치파일의 경로는 다음과 같으며, 패치서버에도 벤더의 FTP와 유사한 디렉터리 구조를 유지한다.

/pub/redhat/linux/updates/9/en/os/SRPMS/kernel-2.4.20-30.9.src.rpm

실질적으로 패치파일이 존재하는 updates/ 디렉터리를 루트로 설정하고, 그 뒤 경로를 패치서버와 동기화 시킨다. 다음과 같은 경로로 패치서버에 저장되며, 이 경로는 DB에 저장된다.

/linux/9/en/os/SRPMS/kernel-2.4.20-30.9.src.rpm (2)

벤더의 분배서버와 패치서버의 경로를 동기화시키는 주 목적은 디렉터리를 정의문으로 사용하여 검색을 용이하게 하고, DB를 접근하지 않고 패치파일이 존재하는지를 판별하기 위해서이다.

3.3 자동 업데이트 검색 알고리즘

자동업데이트 검색은 새로운 패치서버를 구성하는 것은 물론, 기구성된 패치서버에 새로운 패치파일을 찾아오는 기능을 해야한다. 그러기 위해서, 벤더에서 제공하는 모든 파일을 검색하여야만 하고, 이를 충족시키기위해 Recursive Call 방식을 사용하였다.

4. DB 자동구성

새로운 패치가 추가되었을 경우, 패치파일의 MD5 체크섬 값과 마지막 생성날짜, 적용가능한 운영체제의 버전, 지원하는 언어체계, 적용되는 프로그램 등 패치파일에 관련된 여러 가지 정보를 DB에 저장하여야 한다. 이 정보들은 패치파일에 대한 정의문이 되며, 자동분배 모

툴이 클라이언트에게 패치를 제공하기 위해 사용된다.

4.1 HTTP 자동업데이트 정의문

HTTP 자동업데이트를 사용하여 정의문을 생성하는 방법은 문서의 파싱된 데이터에 의존한다. HTTP문서에서 추출된 링크는 (1)과 같으며, 파일이름 WindowsXP- 의 부분에서 운영체제의 버전을 알아내고, KB824146에서 버전을 알아낸다. 그리고, 파일의 최종 수정된 날짜 정보를 이용해 게시날짜를 찾는다. 파일이름에 관련정보가 표시되지 않을 경우, 다운로드 되는 파일이 검색된 HTTP문서에서 'System Requirements' 혹은 '지원 되는 운영 체제'와 같은 키워드로 검사하여 정의문을 생성한다. 마찬가지로, 버전과 게시날짜 등은 'version' 혹은 ' 버전', 'Date Published' 혹은 '게시 날짜'와 같은 키워드로 검색하여 정의문을 생성한다.

4.2 FTP 자동업데이트 정의문

FTP 자동업데이트에서 생성된 패치파일은 HTTP 자동업데이트의 패치파일과는 다르게, 경로를 가지고 있으므로 이것을 이용하여 정의문을 생성한다. 생성되는 패치파일은 위에서 예로 든 (2)와 같으며 경로와 파일이름을 파싱하여 정의문을 생성한다. 실험에서 생성된 정의문은 9, os, kernel, 2.4.20이며, 이것은 각각 운영체제 버전, 적용프로그램, 정의문, 버전을 뜻한다. 자동분배 모듈에서 DB 검색 정의문은 LIKE %를 사용하여 검색하고, 정의문의 버전 추출 토큰은 '/'와 '-'를 사용하였다.

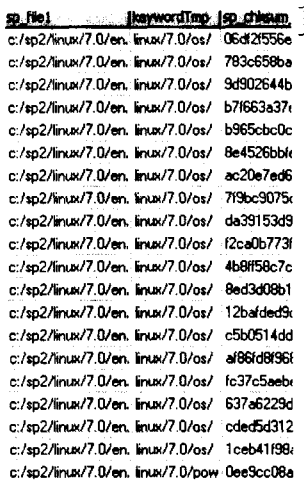
5. 자동업데이트를 이용한 DB 자동구성 구현

제안된 솔루션은 윈도우 환경에서 Java로 구현되었으며, 데이터베이스는 MySQL을 사용하였다. 벤더 서버에 접속하여 패치파일을 검색하고, 패치 분배서버에 저장, 데이터베이스 자동구성과정을 수행한다. 이 솔루션은 비단, 벤더뿐만 아니라 같은 방식으로 구성된 타 패치 분배서버에도 적용 가능하다.

[그림 4] 자동구성 모듈의 실행

6. 결론

이 논문에서 제안하는 자동업데이트를 이용한 패치DB 자동구성방안은 관리자의 불필요한 업무를 줄이고, 모든 네트워크에 패치 파일을 벤더레벨에서 제공하는 것과 같은 효과가 있다. 이것은 관리자측면의 편리성과 더불어,



[그림 5] 자동구성된 DB [그림 6] 자동구성된 서버

클라이언트 측면의 패치 파일에 대한 높은 신뢰도와 정확성을 제공한다.

그러나, 벤더사에서 제공하는 게시물들의 형태가 일정하지 않다는 점과 게시물의 이동이 빈번하게 이루어지고, 존재하지 않는 링크와 쓰레기링크의 추출 등을 통한 패치 검색의 성능이 비효율적일 수 있다. 그리고, 기구성된 DB에 새로운 패치 추가시, 새로운 DB를 구성하는 로직과 같은 검색 로직을 수행하므로 새로운 패치를 추가할 때 소모되는 시간은 상당히 비효율적이다.

기구성된 DB에서 패치를 추가하는 새로운 알고리즘의 제작과 벤더에서 제공하는 통일되지 않은 형태의 HTTP 문서를 정확히 파싱하는 포괄적인 룰의 정의가 향후 정리되어야 할 문제이다.

7. 참고문헌

[1] 손태식 외 "안전한 보안패치 분배 구조의 설계 및 구현", 한국정보보호학회, 논문지, Vol.13, No.4, 08/ 2003

[2] LLNL, SafePatch, Lawrence Livermore National Laboratory

[3] "Enterprise security for Linux",IBM, Tivoli software

[4] C. Kaufman, "DASS(Distributed Authentication Security Service)", Network Working Group Request for Comments: 1507 , September 1993

[5] MicroSoft, "Software Update Services Diploy Guide", http://www.microsoft.com/korea/windows2000/docs/SUSDeployguid.doc

[6] MicroSoft, "Software Update ServiceComponents and Features", http://www.microsoft.com/windowsserv ersystem/sus/suscomponents.mspx

[7] Redhat, "Red hat Network Overview", http://www.redhat.com/whitepapers/rhn/INS0003US_R HN_collateral.pdf

[8] Quick Guide to Red Hat's Package Manager, http://www.tfug.org/helpdeL나/linux/rpm.html